Reverse Engineering Report – Crackme Challenge

Author: Ali Rashchi Github: AliMMA17 Date: 2025-08-10

Legal & Ethical Considerations

This work was performed exclusively for educational purposes on a publicly available crackme challenge designed for reverse engineering practice. The patched binary will not be redistributed. All modifications and analysis respect the platform's guidelines.

Contact: alirashchi7@gmail.com

Challenge information

Challenge: **https://crackmes.one/crackme/5c6fb03b33c5d4776a837d14**

Title: **Funny_Gopher's StupidCrackMe**

Platform: **Windows**

Difficulty: **1.0**

Language: **C/C++**

Objective

The objective of this challenge was to bypass the password validation check within the target executable and achieve the success state without providing the correct password.

Tools Used

- x32dbg
- CFF Explorer
- Windows 11 CMD

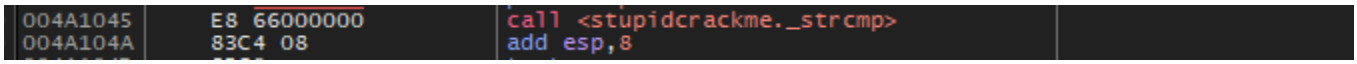
Methodology

1. Loaded the crackme in x32dbg and located the call to strcmp where the input password is compared to the hardcoded password.
2. Found a free spot in the .text section of the binary (executable memory) and wrote:
   xor eax, eax ; force strcmp to return 0 (strings match)
   jmp <return_address>
3. Modified the original call to strcmp to jump to the custom code in .text.
4. Verified the patch by running the program and entering any password, which resulted in the success message being displayed.

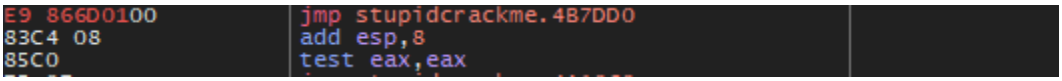
Screenshots and Addresses

Below are the screenshots showing the key steps in the crack:

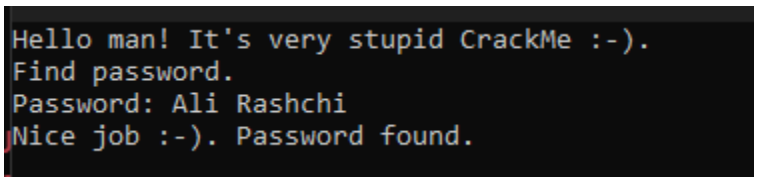
Added instructions in free memory



Before patching



After patching



Output Result

Results

The patched executable successfully bypasses the password validation routine, allowing access to the success message regardless of the entered password. The patch was performed in the .text section because it is executable memory, unlike .rdata which is read-only.